

1 **CLAIMS:**

2  
3 1. A method for generating a delta between a first program binary and  
4 a second program binary, the method comprising the steps of:

5 obtaining a first control flow graph (CFG) representation of the first binary  
6 and obtaining a second CFG representation of the second binary;

7 comparing the first and second CFG representations to identify blocks  
8 (nominally matched blocks) that match in the first and second CFG  
9 representations, thereby identifying blocks (nominally unmatched blocks) in the  
10 second CFG representation that do not match in the first CFG representation;

11 determining edit-operations that merges the unmatched blocks into the first  
12 CFG representation so that first CFG representation is substantially identical to the  
13 second CFG representation;

14 producing a delta comprising the unmatched blocks and the edit-operations.  
15

16 2. A method for transmitting a delta comprising:

17 a method for generating a delta as recited in claim 1;

18 transmitting the delta over a network.  
19

20 3. A method for patching a copy of the first program binary, the  
21 method comprising:

22 a method for generating a delta as recited in claim 1;

23 patching the copy of the first program binary so that the copy is  
24 substantially identical to the second program binary, wherein the delta guides such  
25 patching.

1  
2       4.     A method as recited in claim 1, wherein the comparing step further  
3 comprises matching blocks across the first and second CFG representations based,  
4 at least partially, upon content of blocks being compared and neighborhoods of  
5 blocks local to blocks being compared.

6  
7       5.     A method as recited in claim 1, wherein the comparing step further  
8 comprises matching blocks across the first and second CFG representations based,  
9 at least partially, upon content of the blocks being compared and a local  
10 neighborhoods of blocks surrounding the blocks being compared, wherein a local  
11 neighborhood of a block is augmented with a random sampling of blocks from a  
12 substantially large neighborhood of blocks surrounding the block.

13  
14       6.     A computer-readable medium having embodied thereon a data  
15 structure, comprising a delta generated in accordance with the steps recited in  
16 claim 1.

17  
18       7.     A computer-readable medium having computer-executable  
19 instructions that, when executed by a computer, performs the method as recited in  
20 claim 1.

1           8.     A method for matching blocks between a first control flow graph  
2 (CFG) representation of a portion of a first program and a second CFG  
3 representation of a portion of a second program, the method comprising:

4                 matching blocks between the first and second CFG representations based  
5 upon the content of the blocks;

6                 detecting outliers, wherein outliers are blocks in the first CFG  
7 representation that do not match any block in the second CFG representation  
8 during the matching step;

9                 computing a neighborhood of each block in the first and second CFG  
10 representations by performing a breadth first traversal;

11                 removing the outliers from each neighborhood.

12  
13           9.     A method as recited in claim 8 further comprising:

14                 computing labels for each block in first and second CFG representations  
15 based upon content of a block;

16                 for each neighborhood computed in the computing step, forming a “d-  
17 label” for each block in a neighborhood based upon the labels of the blocks within  
18 the neighborhood;

19                 d-label matching blocks between first and second CFG representations  
20 based upon the d-label of the blocks.

21  
22           10.    A computer-readable medium having computer-executable  
23 instructions that, when executed by a computer, performs the method as recited in  
24 claim 8.  
25

1        **11.** A method for matching procedures between a first control flow  
2 graph (CFG) representation of a portion of a first program and a second CFG  
3 representation of a portion of a second program, the method comprising:

4            computing a procedure-match-criterion for each procedure in the second  
5 CFG representation, where the procedure-match-criterion for a procedure in the  
6 second CFG representation represents the number of matching blocks between that  
7 procedure and a specified procedure in the first CFG representation;

8            matching procedures in the second CFG representation with the specified  
9 procedure in the first CFG representation based upon the procedure-match-criteria  
10 for the procedures in the second CFG representation.

11  
12        **12.** A computer-readable medium having computer-executable  
13 instructions that, when executed by a computer, performs the method as recited in  
14 claim 11.

15  
16        **13.** A method for matching of blocks in a procedure of a first control  
17 flow graph (CFG) representation of a portion of a first program between an  
18 ostensibly matching procedure of a second CFG representation of a portion of a  
19 second program, the method comprising:

20            matching blocks between the first and second CFG representations based  
21 upon the content of the blocks;

22            computing successively smaller neighborhoods of each block in the first  
23 and second CFG representations via breadth first traversals;

1       for each neighborhood computed in the computing step, forming a “d-  
2 label” for each block in a neighborhood based upon the labels of the blocks within  
3 the neighborhood;

4       d-label matching blocks between first and second CFG representations  
5 based upon the d-label of the blocks.

6  
7       **14.**   A method as recited in claim 13, wherein at least one neighborhood  
8 computed in the computing steps is augmented with a random sampling of blocks  
9 in the complete representation of the neighborhood.

10  
11       **15.**   A computer-readable medium having computer-executable  
12 instructions that, when executed by a computer, performs the method as recited in  
13 claim 13.

14  
15       **16.**   A method for facilitating matching of blocks between a first control  
16 flow graph (CFG) representation of a portion of a first program and a second CFG  
17 representation of a portion of a second program, whereby blocks may match  
18 except for nominal differences, the method comprising:

19       computing register flows for each block by analyzing a def-use chain and a  
20 use-def chain;

21       for each block, assigning def-identifiers to definitions (“def”s) of each  
22 register;

23       for each block, associating a use of a register in a block with the def  
24 identifier of all defs of that register that reach this use;  
25

1 matching blocks between the first and second CFG representations based  
2 upon the def-identifiers and use-identifiers.

3  
4 17. A computer-readable medium having computer-executable  
5 instructions that, when executed by a computer, performs the method as recited in  
6 claim 16.

7  
8 18. A patch data structure generated in accordance with the following  
9 acts:

10 providing a server computer in a communications with a communications  
11 network;

12 receiving input from a client computer by way of the communications  
13 network, the input providing a parameter indicative of a request for upgrading a  
14 copy of a first program binary to a match a second program binary;

15 retrieving a delta between a first program binary and the second program  
16 binary, wherein computing such delta comprises the steps of:

17 a) obtaining a first control flow graph (CFG) representation of the first  
18 binary and obtaining a second CFG representation of the second  
19 binary;

20 b) comparing the first and second CFG representations to identify  
21 blocks (nominally matched blocks) that match in the first and second  
22 CFG representations, thereby identifying blocks (nominally  
23 unmatched blocks) in the second CFG representation that do not  
24 match in the first CFG representation;

1 c) determining edit-operations that merges the unmatched blocks into  
2 the first CFG representation so that first CFG representation is  
3 substantially identical to the second CFG representation;  
4 d) producing a delta comprising the unmatched blocks and the edit-  
5 operations;  
6 generating the patch data structure as a function of the delta.

7  
8 **19.** A method for transmitting a patch data structure comprising  
9 transmitting a patch data structure as recited in claim 18 over a communications  
10 network.

11  
12 **20.** A method for patching a copy of the first program binary at a client  
13 computer, the method comprising patching the copy of the first program binary so  
14 that the copy is substantially identical to the second program binary, wherein a  
15 delta in a patch data structure as recited in claim 18 guides such patching.  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25

1        **21.**    A delta-generator system, comprising:

2        a comparator that is configured to compare a first control flow graph (CFG)  
3 representation of a first program binary and a second CFG representation of the  
4 second program binary for identifying blocks (nominally matched blocks) that  
5 match in the first and second CFG representations, thereby identifying blocks  
6 (nominally unmatched blocks) in the second CFG representation that do not match  
7 in the first CFG representation;

8        an edit-op determiner configured to determine the edit-operations that  
9 merges the unmatched blocks into the first CFG representation so that first CFG  
10 representation is substantially identical to the second CFG representation;

11       an output sub-system that is configured to produce a delta comprising the  
12 unmatched blocks and the edit-operations.

13  
14       **22.**    A computer-readable medium having embodied thereon a data  
15 structure comprising a delta produced by the system as recited in claim 21.  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25